



500.43289X00

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s): S. HORIKAWA

Serial No.: 10/714,597

Filed: November 18, 2003

Title: JOB SCHEDULING MANAGEMENT METHOD, SYSTEM, AND PROGRAM

LETTER CLAIMING RIGHT OF PRIORITY

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

December 10, 2003

Sir:

Under the provisions of 35 USC 119 and 37 CFR 1.55, the applicant(s) hereby claim(s) the right of priority based on:

Japanese Patent Application No. 2003-193259
Filed: July 8, 2003

A certified copy of said Japanese Patent Application is attached.

Respectfully submitted,

ANTONELLI, TERRY, STOUT & KRAUS, LLP

Carl I. Brundidge
Registration No.: 29,621

CIB/rr
Attachment

日 本 国 特 許 庁
JAPAN PATENT OFFICE

別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office.

出 願 年 月 日 2 0 0 3 年 7 月 8 日
Date of Application:

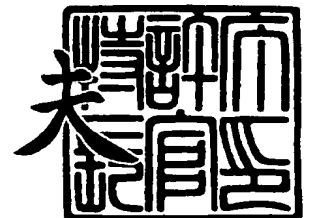
出 願 番 号 特 願 2 0 0 3 - 1 9 3 2 5 9
Application Number:
[ST. 10/C] : [J P 2 0 0 3 - 1 9 3 2 5 9]

出 願 人 株式会社日立製作所
Applicant(s):

特許庁長官
Commissioner,
Japan Patent Office

2 0 0 3 年 1 1 月 1 1 日

今 井 康 夫



【書類名】 特許願

【整理番号】 K03004391A

【あて先】 特許庁長官殿

【国際特許分類】 G06F 9/46

【発明者】

【住所又は居所】 神奈川県横浜市戸塚区戸塚町5030番地 株式会社日立製作所 ソフトウェア事業部内

【氏名】 堀川 茂

【特許出願人】

【識別番号】 000005108

【氏名又は名称】 株式会社日立製作所

【代理人】

【識別番号】 100075096

【弁理士】

【氏名又は名称】 作田 康夫

【手数料の表示】

【予納台帳番号】 013088

【納付金額】 21,000円

【提出物件の目録】

【物件名】 明細書 1

【物件名】 図面 1

【物件名】 要約書 1

【プルーフの要否】 要

【書類名】 明細書

【発明の名称】 ジョブスケジューリング管理方法及びシステム並びにプログラム

【特許請求の範囲】

【請求項 1】

ネットワークを介して接続される計算機に割り当てるジョブのスケジュールを管理するジョブスケジュール管理方法であって、前記ジョブが割り当てられた計算機の稼動状況を監視し、前記稼動状況が所定の条件を満たすか否か判断し、稼動状況が所定の条件を満たす場合、前記計算機に割り当てられたジョブのうち、前記所定の条件を満たした時点で未完了のジョブを検出し、当該検出した未完了のジョブの実行に必要なリソース情報に基づいて、当該検出した未完了のジョブを実行可能な他の計算機を抽出し、当該検出した未完了のジョブを前記抽出した他の計算機に割り当てることを特徴とするジョブスケジュール管理方法。

【請求項 2】

前記所定の条件を満たすか否かの判断は、CPU使用率が所定の使用率を超えた回数により行うことを特徴とする請求項 1 記載のジョブスケジュール管理方法。

【請求項 3】

ネットワークを介して接続される複数の計算機にジョブを割り当て、当該ジョブのスケジュールを管理する管理計算機におけるジョブスケジュール管理方法であって、前記ジョブと当該ジョブを割り当てた計算機との対応付けを示す第 1 の情報と、前記ジョブの実行に必要なリソースを示す第 2 の情報と、各計算機で使用可能なリソースを示す第 3 の情報とを管理し、前記ジョブを割り当てた計算機の稼動状況を監視し、前記稼動状況が所定の条件を満たすか否か判断し、当該判断結果に応じて、前記計算機に割り当てたジョブのうち未完了のジョブを前記第 1 の情報を用いて検出し、当該検出した未完了のジョブの実行に必要なリソースを前記第 2 の情報を用いて抽出し、当該抽出されたリソースを使用可能な他の計算機を前記第 3 の情報を用いて抽出し、前記検出した未完了のジョブを前記抽出した他の計算機に割り当てることを特徴とするジョブスケジュール管理方法。

【請求項 4】

前記検出した未完了のジョブを前記抽出した他の計算機に割り当てる際に、前記他の計算機に既に割り当てているジョブを含めて再スケジューリングを行うことを特徴とする請求項 3 記載のジョブスケジュール管理方法。

【請求項 5】

前記検出した未完了のジョブを前記抽出した他の計算機に割り当てる際に、前記抽出した他の計算機に既に割り当てているジョブのうち未完了のジョブを前記第 1 の情報を用いて検出し、当該検出した未完了のジョブの実行に必要なリソースを前記第 2 の情報を用いて抽出し、当該抽出されたリソースを使用可能な更に他の計算機を前記第 3 の情報を用いて抽出し、前記検出した未完了のジョブを前記抽出した更に他の計算機に割り当てることを特徴とする請求項 3 記載のジョブスケジュール管理方法。

【請求項 6】

前記管理計算機は、自計算機に対してジョブを割り当てることを特徴とする請求項 3 記載のジョブスケジュール管理方法。

【請求項 7】

前記管理計算機は、前記ジョブと当該ジョブの終了が必要な時間との対応付けを示す情報と、前記ジョブの実行に有する時間を示す情報とをさらに管理し、前記ジョブが予め定められた前記ジョブの終了が必要な時間に終了しないことを、前記ジョブを実行する計算機の稼動状況と前記ジョブの実行に有する時間から予測した場合は、前記所定の条件を満たさないとして、前記計算機に割り当てたジョブのうち未完了のジョブを他の計算機に割り当てることを特徴とする請求項 3 記載のジョブスケジュール管理方法。

【請求項 8】

前記検出した未完了のジョブを前記抽出した他の計算機に割り当てる際、前記検出した未完了のジョブを、当該ジョブの実行に必要なリソースに応じて、複数の他の計算機に割り当てることを特徴とする請求項 3 記載のジョブスケジュール管理方法。

【請求項 9】

ネットワークを介して接続される複数の計算機にジョブを割り当て、当該ジョブのスケジュールを管理する管理計算機であって、前記複数の計算機のうち第1の計算機に第1のジョブを割り当て、第2の計算機に第2のジョブを割り当てたことを示す情報を管理する管理手段と、前記第1の計算機の稼動状況を監視する監視手段と、当該監視手段からの指示に応じて、前記管理手段で管理する情報のうち前記第1の計算機に割り当てた第1のジョブを前記第2の計算機に割り当て直し、前記第2の計算機に割り当てた第2のジョブを第3の計算機に割り当て直す再スケジュール手段とを有することを特徴とするジョブスケジュール管理計算機。

【請求項10】

ネットワークを介して接続される複数の計算機にジョブを割り当て、当該ジョブのスケジュールを管理する管理計算機で用いられるジョブスケジュール管理プログラムであって、前記ジョブと当該ジョブを割り当てた計算機との対応付けを示す情報と、前記ジョブの実行に必要なリソースを示す情報と、各計算機で使用可能なリソースを示す情報とを管理する機能と、前記ジョブを割り当てた計算機の稼動状況を監視する機能と、前記稼動状況が所定の条件を満たすか否か判断する機能と、当該判断結果に応じて、前記計算機に割り当てたジョブのうち未完了のジョブを検出する機能と、当該検出した未完了のジョブの実行に必要なリソースを抽出する機能と、当該抽出されたリソースを使用可能な他の計算機を抽出する機能と、前記検出した未完了のジョブを前記抽出した他の計算機に割り当てる機能とを備えることを特徴とするジョブスケジュール管理プログラム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、ジョブの運用管理を行なうためのジョブ運用管理システムに関するものであり、特にジョブスケジュール管理方法及びシステム並びにプログラムに関する。

【0002】

【従来の技術】

コンピュータが行う複数の処理のうち、関連する一連の処理を纏めて1つのジョブとみたと、複数のジョブをスケジュールに従って実行させる技術がある。このような技術において障害が発生した場合、できるだけ早急にジョブを復帰させる必要がある。このような場合、同一ジョブが実行可能な複数の計算機を予め特定しておき、ジョブを再実行させる際には、前記予め特定された計算機のうち障害回復に最適な計算機を用いる技術がある（例えば、特許文献1参照）。

【0003】

また、複数のジョブステップからなり、ジョブステップ間で依存関係があるジョブに対して、当該複数のジョブステップ全てを実行可能な複数の計算機を予め特定しておき、複数のジョブステップを実行する際、単一の計算機により実行するのではなく、ジョブステップ実行中に後続のジョブステップを実行するに最適な計算機を前記予め特定された複数の計算機の中から選定し、実行する技術がある（例えば、特許文献2参照）。

【0004】

また、予め各計算機と固定的に対応付けて登録された各ジョブに対して、障害発生前に実行済みでないジョブの処理終了予定時刻を算出し、この処理終了予定時刻が、各ジョブの終了すべき時刻より前であるジョブについてのみを対象として、各ジョブの優先度に従って抽出された順に実行順序を付与してスケジュールを再構成し、再構成されたスケジュールによりジョブを実行する技術がある（例えば、特許文献3参照）。

【0005】

【特許文献1】

特開平11-353284号公報

【特許文献2】

特開2001-166956号公報

【特許文献3】

特開2001-350673号公報

【0006】

【発明が解決しようとする課題】

上記従来技術は、障害発生時や処理負荷集中時、特定のジョブを実行可能な計算機として、特定のジョブと予め固定的に対応付けて認識されている特定の計算機を用いて代替処理を行うもので、予め特定のジョブを実行可能な計算機として対応付けられている特定の計算機の中でジョブスケジュールを再構成するものであり、ジョブの性質等（例えば、ジョブに応じて異なる実行に必要なリソース）とその稼動状況に応じて、スケジューリングの際に適宜代替となる計算機を抽出・選択するものではない。そのため、スケジュールを再構築する際に対象となるジョブは、障害ジョブやそれと依存関係があるジョブのみである。

【0007】

また、障害ジョブやそれと依存関係があるジョブのみを考慮してスケジュールを再構築しているため、代替となる計算機のジョブ割り当て状態（例えば、既に割り当てられている依存関係のないジョブ）を考慮して、スケジュールを再構成することについては開示されていない。

【0008】

本発明は、上記課題を考慮したジョブ運用管理技術を提供することを目的とする。

【0009】

本発明は、ジョブを実行する計算機で障害や性能の劣化が発生した場合に他の計算機や他のジョブのスケジュールを考慮したジョブのスケジュールを再構成する技術を提供することを目的とする。

【0010】

【課題を解決するための手段】

本発明は、ネットワークを介して接続される計算機（例えば、実行サーバ）や、計算機で実行されるエージェントプログラムに割り当てるジョブのスケジュールを管理する管理計算機（管理専用サーバであっても、実行サーバと兼用であっても良い）におけるジョブスケジュール管理方法であって、前記ジョブが割り当てられた計算機や当該計算機が利用できるリソースの稼動状況を監視したり異常を検知したりし、前記稼動状況が所定の条件を満たすか否かを判断し（例えば、CPU使用率、使用メモリ量、空きディスク容量が十分か否かの判断や、障害発生

を検知したかどうかの判断等)、稼動状況が所定の条件を満たす場合、前記計算機に割り当てられたジョブのうち未完了のジョブ(ジョブが未着手のもの、ジョブが終了していないもの、ジョブの実行が失敗したもの等)を検出し、当該検出した未完了のジョブの実行に必要なリソース情報(CPU、メモリ、ディスク、データベース等)に基づいて、当該検出した未完了のジョブを実行可能な他の計算機を抽出し、当該検出した未完了のジョブを前記抽出した他の計算機に割り当てることを特徴とする。

【0011】

また、ジョブと当該ジョブを割り当てた計算機との対応付けを示す第1の情報と、前記ジョブの実行に必要なリソースを示す第2の情報と、各計算機で使用可能なリソースを示す第3の情報とを管理し、前記ジョブを割り当てた計算機の稼動状況を監視し、前記稼動状況が所定の条件を満たすか否か判断し、当該判断結果に応じて、前記計算機に割り当てたジョブのうち未完了のジョブを前記第1の情報をを用いて検出し、当該検出した未完了のジョブの実行に必要なリソースを前記第2の情報をを用いて抽出し、当該抽出されたリソースを使用可能な他の計算機を前記第3の情報をを用いて抽出し、前記検出した未完了のジョブを前記抽出した他の計算機に割り当てることを特徴とする。

【0012】

ここで、前記検出した未完了のジョブを前記抽出した他の計算機に割り当てる際に、前記他の計算機に既に割り当てているジョブを含めて再スケジューリングを行っても良い。

【0013】

また、前記検出した未完了のジョブを前記抽出した他の計算機に割り当てる際に、前記抽出した他の計算機に既に割り当てているジョブのうち未完了のジョブを前記第1の情報をを用いて検出し、当該検出した未完了のジョブの実行に必要なリソースを前記第2の情報をを用いて抽出し、当該抽出されたリソースを使用可能な更に他の計算機を前記第3の情報をを用いて抽出し、前記検出した未完了のジョブを前記抽出した更に他の計算機に割り当てても良い。

【0014】

また、前記管理計算機は、前記ジョブと当該ジョブの終了が必要な時間との対応付けを示す情報と、前記ジョブの実行に有する時間を示す情報とをさらに管理し、前記ジョブが予め定められた前記ジョブの終了が必要な時間に終了しないことを、前記ジョブを実行する計算機の稼動状況と前記ジョブの実行に有する時間から予測した場合は、前記所定の条件を満たさないとして、前記計算機に割り当てたジョブのうち未完了のジョブを他の計算機に割り当てても良い。

【0015】

また、前記検出した未完了のジョブを前記抽出した他の計算機に割り当てる際、前記検出した未完了のジョブを、当該ジョブの実行に必要なリソースに応じて、複数の他の計算機に割り当てても良い。

【0016】

尚、上記目的を達成するためには、上述した機能を実現するシステム、プログラム若しくはプログラムを格納した記録媒体であっても良い。

【0017】

【発明の実施の形態】

以下本発明の実施の形態を、図面を参照しながら詳細に説明する。図1は、ジョブを実行するシステム構成の一実施形態を示す全体構成図である。

【0018】

本システムは、ネットワーク130を介してジョブの運用管理を行う管理計算機（本実施例では、サーバ）100、ジョブの実行をする実行計算機（本実施例では、サーバ1、サーバ2、サーバ3；101）が接続されている。実行サーバ101は1台であっても、複数であってもよいが、実行サーバで実行されるジョブに依存関係がない場合は、システム全体として複数のエージェントを有することとなる。

【0019】

管理計算機（サーバ）100は、専用計算機（サーバ）であっても、実行計算機（サーバ）101と併用であっても良い。本実施形態では、管理計算機（サーバ）100を専用計算機（サーバ）とし、情報を表示する表示部やキーボード・マウス等の入力部等からなる入出力部を有し、ジョブ処理システムの操作を行う

コンソール 103 が接続されている。実行計算機（サーバ）101 には、各実行サーバで利用可能なリソース 1、リソース 2（102）が接続されている。

【0020】

ここで、リソースとは、CPU、メモリ、データベース、ディスク装置、ストレージなどのジョブ実行に利用されるものであり、稼動性能を取得できるものであれば良く、本実施例では、リソース 1、リソース 2（102）が実行計算機（サーバ）101 に対して外付けで表示されているが、実行計算機（サーバ）101 内にあるものでも良い。また、リソース 1、リソース 2（102）と実行計算機 101 との間を SAN（Storage Area Network）環境で構築可能である。本発明はフェイルオーバーやディザスタリカバリの際のスケジュール管理に適用可能である。

【0021】

管理計算機（サーバ）100 と実行計算機（サーバ）101 は、大型コンピュータやサーバシステムなどで、汎用のコンピュータであっても、専用のハードウェアにより構成されているものであってもよく、システム全体の制御及びプログラム処理を行うマイクロチップなどの処理部（制御部）（111、122）と、プログラムやデータを保持する記憶部（記憶装置）（112、123）とを含んで構成されている。

【0022】

記憶部（112、123）はメモリ等の主記憶装置やハードディスク等の補助記憶装置、データベース等から構成され、記憶部 112 にはマネージャプログラムの他、管理サーバ 100 の処理上保持すべきデータやプログラムが記憶されている。記憶部 123 にはエージェントプログラム、アプリケーションプログラムの他、実行サーバ 101 の処理上保持すべきデータやプログラムが記憶されている。

【0023】

管理サーバ 100 の機能と実行サーバ 101 の機能とは、その用途に応じて、別個独立のシステムとして管理されても、一体化されたシステムとして提供されても良い。また、記憶部 112、123 の機能の一部を、管理サーバ 100 や実

行サーバ101の外部に存在する固有データベースや共通データベース等として管理しても良い。

【0024】

記憶部(112、123)に格納されたプログラム等と処理部(111、122)とが連動して、管理サーバ100では、ジョブ自身やジョブを管理するジョブ処理システムのマネージャ110、実行サーバ101では、ジョブ自身や、ジョブを実行するリソースや実行サーバで動作しているアプリケーション121の稼動性能を収集するジョブ処理システムのエージェント120を実行する。アプリケーション121は、データベースアプリケーションなどの一般のアプリケーションである。

【0025】

各実行サーバの稼動性能や利用できるリソース、アプリケーションの種類は異なる場合がある。

【0026】

図2は、マネージャ110の主要機能の一実施形態を示す機能ブロック図である。

【0027】

マネージャ110は、実行サーバ101やエージェント120に割り当てるジョブのスケジュールを管理するスケジュール管理部200、管理サーバ100が管理している実行サーバ101上のエージェント120を管理するエージェント管理部201、各実行サーバから利用できるリソース102や各実行サーバ上のアプリケーション121や処理部122を管理するリソース管理部202、現在ジョブを実行しているエージェント120を管理するジョブ実行中エージェント管理部203、実行サーバ101で障害が発生したり、性能が劣化した場合にジョブのスケジュールの再構成を行う再スケジューリング部204、ジョブの実行に必要なリソースを有する実行サーバ101や当該実行サーバに存在するエージェントを検索する代替エージェント検索部205、マネージャ110や他のエージェント120と通信するための通信部206とを有する。

【0028】

マネージャ110は、管理サーバ100の記憶部112に、ジョブの実行される時間や実行される実行サーバの情報を管理しているスケジュールテーブル300（後述）、実行サーバ101で実行されるジョブを管理したジョブテーブル400（後述）、マネージャの管理しているエージェントの情報を管理したエージェント管理テーブル500（後述）、またジョブ処理システム内で利用可能な全てのリソースとそれを使用できる実行ノードの関連を管理しているリソース間距離テーブル600（後述）を適宜格納したり、前記テーブルに格納されるデータを参照することによって本実施形態を実現する。

【0029】

図3は、スケジュールテーブル300のデータ構成の一例を示す図である。

【0030】

このスケジュールテーブル300は、管理者によって設定され、再スケジュールの際にはマネージャ110によって変更される。マネージャ110のスケジュール管理部200は、スケジュールテーブル300を常に監視し、このテーブルに従って通信部206を介してジョブの実行命令を実行サーバ101に伝える。ジョブの実行命令を伝えた後、ジョブ実行中エージェント管理部203が実行サーバ101上のエージェントの監視を開始する。

【0031】

スケジュールテーブル300は、ジョブの運用を管理するための情報である。具体的には、実行開始日時301、最終終了日時302、最大実行時間303、ジョブを実行する実行エージェント304、実行ジョブ305に関する情報が含まれる。実行開始日時301は、ジョブを開始する日時を示し、実行を行う日、時間を含む。定期的に実行するジョブであるならば、その起動時刻を示す。最終終了日時302は、ジョブが最終的に終了していなければならない時間や最終的に終了していて欲しい時間を示す。特に制限がない等の場合は、必ずしも設定される必要はない。最大実行時間303は、ジョブの最大実行時間や期間を示し、ジョブの実行時間がこの値を超える場合を、マネージャ110によるスケジュールの再構成のきっかけとすることができる。実行エージェント304は、ジョブを実行するために実行サーバ101に存在するエージェントを示す。実行ジョブ

305は、実行されるジョブを示す。

【0032】

図4は、ジョブテーブル400のデータ構成の一例を示す図である。

【0033】

ジョブテーブル400は、管理者が設定するテーブルで、ジョブの実行内容を管理するテーブルであり、実行に必要な情報を含む。マネージャ110のスケジュール管理部200は、スケジュールテーブル300から実行するジョブを選択し、そのジョブの内容をこのテーブルから取得する。

【0034】

具体的には、ジョブ名401、実行内容402、実行必要リソース403、最大許容コスト404、優先度405に関する情報を含む。ジョブ名401は、ジョブの名称を示す。実行内容402は、そのジョブが実行する処理内容を示す。実行必要リソース403は、そのジョブを実行するために必要なリソース（CPU、メモリ、ディスク、データベースソフト、その他のリソース1、その他のリソース2等）を表す。例えば、データベースアプリケーションソフトとディスク装置が必要なジョブならば、それらの必要なリソースの条件が設定される。最大許容コスト404は、ジョブを1度実行する時に消費して良い最大のコストを示す。

【0035】

ここで、コストとは、リソースを使用する場合の効率を表す値である。この値が小さいほどジョブの実行を効率良く行う事ができることを示す。コストの値は、例えば、ジョブを実行するために必要なリソースの使用単位コストの和から算出する。優先度405は、各ジョブの優先度を示す。本実施形態では、優先度の高いものをA、低いものをCとしている。

【0036】

図5は、エージェント管理テーブル500のデータ構成の一例を示す図である。

【0037】

エージェント管理テーブル500は、マネージャ110が管理するエージェン

ト 120 に関する情報を管理するテーブルであり、エージェントが実行されている実行サーバ 101 との関連を示す。

【0038】

具体的には、エージェント名 501 は、エージェントの名称を示す。マネージャ 110 のスケジュール管理部 200 は、スケジュールテーブル 300 から実行するエージェント 120 を取得し、そのエージェント 120 が動作している実行サーバ 101 を取得する。例えば、エージェント 2 とエージェント 4 とが実行サーバ 2 に対応付けられているように、1 台の実行サーバ 101 に複数のエージェント 120 を設定することができる。複数のエージェントが存在する実行サーバは、複数のジョブ（特に依存関係のない複数のジョブ）を並行して処理したりする場合にも対応可能である。

【0039】

図 6 は、リソース間距離テーブル 600 のデータ構成の一例を示す図である。

【0040】

リソース間距離とは、実行サーバ 101 からリソースを使用するときにかかる単位コストを示す。この値は、マネージャ 110 がエージェントから取得した稼働性能を元にリソース管理部 202 を介して随時変更される。本実施形態では、リソースが使用できない場合（リソースが割り当てられていない場合、障害等で一時的に使用できない場合など）、単位コストは表示されていない。

【0041】

リソースが使用できない場合に、そもそもリソースが割り当てられていないのか、障害等で一時的に使用できないのか（どの程度で回復見込みか、使用可能となった場合の単位コストはどの程度か）といった状態が判断可能な情報も保持することによって、スケジュール再構築の際の一判断材料とすることも可能である。

【0042】

具体的には、使用可能リソース 601 は、マネージャ 110 が管理するエージェント 120 が使用可能な全てのリソースを示す。使用実行サーバ 602 は、使用可能リソース 601 を使用できる実行サーバ 101 を示す。例えば、実行サー

バ1のCPUを使用できる実行サーバは、実行サーバ1でその使用単位コストは5であることを示し(603)、逆に実行サーバ2、実行サーバ3では、使用できないことを示す。また、リソース1のディスクは、実行サーバ1と実行サーバ3で使用可能であることを示す(604、605)。

【0043】

またリソースの使用単位コストは、各実行サーバ101上のエージェント120が収集したリソースの稼動性能情報により算出される。例えば、ディスクの単位コストは、実行サーバ101から使用した場合の平均使用時間やI/Oオペレーションの平均実行時間などから算出する。そのため同じリソースを使用した場合でも、使用する実行サーバ101により異なる。単位コストの値が小さいほど実行サーバ101からのリソースの性能が良い事を示す。

【0044】

尚、上述した実施形態ではスケジュールテーブル300とジョブテーブル400とエージェント管理テーブル500とリソース間距離テーブル600とを分けて管理しているが、適宜、一連のデータとして管理しても、データ項目の組み合わせを変更しても良い。

【0045】

図7は、マネージャ110の基本シーケンスの一例を説明するフローチャートである。

【0046】

マネージャ110のエージェント管理部201は、新規のエージェント120からの新規登録要求を受信した場合(S700)、その新規エージェント120を実行サーバと対応付けてエージェント管理テーブル500に登録する(S701)。マネージャ110のジョブ実行中エージェント管理部203は、実行サーバやリソースの稼動状況を監視する手法として、例えば、定期的にエージェントへの問合せを行ったり、障害発生に応じてエージェントから報告を受けたりする。

【0047】

定期的に各エージェント120から稼動性能情報を収集する場合は、パフォーマンス収集時間がきたかどうか判断し(S702)、パフォーマンス収集時間がき

た場合は、管理する全てのエージェント 120 に現在の稼動性能情報を送信するよう要求する(S703)。送信された稼動性能情報を元にリソース管理部 202 を介してリソース間距離テーブルを更新する(S704)。

【0048】

マネージャ 110 のスケジュール管理部 200 は、スケジュールテーブル 300 を検索し、実行時間になったジョブがあった場合(S705)、ジョブの実行命令を送信する(S706)。ジョブ実行命令を送信すると、マネージャ 110 のジョブ実行中エージェント管理部 203 は、ジョブの実行状態とそれを実行しているエージェント 120 の状態を監視することとなる。

【0049】

図 8 は、エージェント 120 の主要機能の一実施形態を示す機能ブロック図である。

【0050】

エージェント 120 は、マネージャ 110 から割り当てられたジョブの実行処理を行うジョブ実行部 800、稼動性能情報の収集を行うパフォーマンス収集部 801、収集した稼動情報から現在の実行サーバ 101 の状態を分析し、ジョブの実行に最適なりソースの使用状態を計算するリソース使用状況管理部 802、マネージャ 110 や他のエージェント 120 と通信を行う通信部 803 とを有する。

【0051】

エージェント 120 は、実行サーバ 101 の記憶部 123 に、収集した稼動性能情報を蓄積するための履歴パフォーマンステーブル 900 (後述)、リソース異常状態を示す条件を保存するリソース使用制限テーブル 1000 (後述) を適宜格納したり、前記テーブルに格納されるデータを参照することによって本実施形態を実現する。

【0052】

エージェント 120 は、マネージャ 110 からジョブの実行命令を受け付けるとジョブ実行部 800 でジョブを実行する。

【0053】

図9は、履歴パフォーマンステーブル900のデータ構成の一例を示す図である。

【0054】

履歴パフォーマンステーブル900は、エージェント120のパフォーマンス収集部801が定期的・不定期的に収集した稼動性能を蓄積する度に更新する。具体的には、収集時刻901、収集項目902から構成される。収集時刻901は、エージェント120が稼動性能を収集した時刻が格納される。収集項目902は、収集した稼動性能を項目毎に格納する。例えば、CPU使用率903の稼動性能が格納される。他にも、使用メモリ量や空きディスク容量、ディスク(平均ディスク処理時間904)やデータベースアプリケーション(平均クエリ処理時間905)などの稼動性能が格納される。

【0055】

図10は、リソース使用制限テーブル1000のデータ構成の一例を示す図である。

【0056】

リソース使用制限テーブル805は、管理者が設定、もしくはエージェント120のリソース使用状況管理部802が更新するテーブルであり、エージェント120のパフォーマンス収集部801が収集した稼動性能から現在のリソースの状態を判定する条件を示す。エージェント120のパフォーマンス収集部801がジョブ実行中に稼動性能を収集する度に判定される。

【0057】

具体的には、ジョブ1001、しきい値1002、しきい値超過回数1003を含んで構成される。ジョブ1001は、適用されるジョブの名称を示す。しきい値1002は、収集した稼動性能からリソースの異常状態かどうかを判断する条件を示す。しきい値1002に指定される条件は、収集する稼動性能の一部を指定する。例えば、ジョブ1は、空きディスク容量が200MB以下である場合にしきい値超過として認識され、カウントされる。しきい値は1つのジョブ1001に対して複数指定できる。しきい値超過回数1003は、しきい値1002を超えた回数を指定する。例えば、ジョブ1は、稼動性能を3回収集したうちの

2 回、しきい値 1002 の条件を超えた場合に、異常状態として認識される。

【0058】

図 11 は、エージェント 120 の基本シーケンスの一例について説明するフローチャートである。

【0059】

エージェント 120 のパフォーマンス収集部 801 は、稼動性能収集時間がきたかどうか判断し (S1100)、定期的に行うサーバ 101 とそれから使用できるリソース 102 の稼動性能を収集し (S1101)、履歴パフォーマンステーブル 900 を更新する。マネージャ 110 からのジョブ実行命令を受信した場合 (S1102)、ジョブ実行部 800 を介してジョブを実行する (S1103)。ジョブの実行中は、常に稼動性能を収集し (S1104)、リソース使用状況管理部 802 を介してリソース使用制限テーブル 1000 の条件を照らし合わせ、実行サーバ 101 もしくはリソース 120 の状態を監視し (S1105)、リソースの状態が異常状態になった場合は、通信部 803 を介してマネージャに異常が発生した事を通知する (S1106)。ステップ 1104 からステップ 1106 の処理をジョブが終了するまで繰り返し、ジョブが終了した場合 (S1107) は、ジョブの使用したリソースとその状態やジョブの最大実行時間 303 とジョブの実際の実行時間から、リソース使用状況管理部 802 を介して次のリソース使用制限を設定する (S1108)。

【0060】

図 12 は、リソース使用制限の設定 (S1108) の一例を示すフローチャートである。リソース使用制限テーブル 1000 に実行したジョブの条件が設定されていない場合 (S1200) と、ジョブが最大終了時間 303 で終了した場合 (S1201) は、最新のジョブが使用したリソースの稼動性能を条件として設定する。それ以外は、リソース使用制限テーブル 1000 の更新は行わない。

【0061】

図 13 は、マネージャ 110 における、実行中のジョブとそれを実行しているエージェント 120 の監視処理のフローチャートである。

【0062】

マネージャ110のジョブ実行中エージェント管理部203は、ジョブが終了するまでエージェント120を監視する(S1300~S1306)。

【0063】

ジョブの実行命令を送信した時やジョブの実行中にエージェントからの応答がない場合等、エージェントで異常が発生した場合(S1300)、エージェント120が使用しているリソースが異常状態(リソースの使用制限の超過やリソース自体の故障)になった場合(S1301)は、実行中のジョブの実行が失敗した(未完了である)ものと判断し、再スケジューリング部204にて実行中のジョブの再スケジューリングを行い、ジョブの再実行を行う(S1302)。ジョブの最大実行時間を経過してもジョブが終了していない場合(S1303)は、実行中のジョブの次に予定されているジョブをスケジュール管理部200を介してスケジュールテーブル300から検索し(S1304)、そのジョブに対してスケジュールの再設定を行う(S1305)。ジョブが終了し(S1306)、ジョブの実行時間が最終終了日時303を超過した場合(S1307)は、ワーニングメッセージをコンソール103に出力し、管理者に伝える(S1308)。

【0064】

図14は、マネージャ110の再スケジューリング部204における処理を示すフローチャートである。

【0065】

再スケジューリングを行う際、マネージャ110の代替エージェント検索部205にて同じジョブを実行できるエージェント120の検索を、ジョブテーブル400やエージェント管理テーブル500やリソース間距離テーブル600を用いて、エージェントが存在する実行サーバから使用可能なリソースに基づいて行う(S1400)。随時リソースの稼動性能情報が反映されるリソース間距離テーブル600を用いて、再スケジュールの際に代替エージェントを検索するので、リソースの稼動状況に応じたジョブの割り当てが可能となる。

【0066】

ここで代替となるエージェント120が1つ以上見つかった場合(S1401

）、そのエージェント 120 の中にジョブの実行する時間帯のスケジュールが空いているエージェントをスケジュールテーブル 300 を用いて検索する（S1402）。スケジュールの空いたエージェントがあった場合（S1402 で「Yes」）、スケジュールテーブル 300 の実行エージェント 304 を変更する（S1411）。

【0067】

スケジュールの空いたエージェント 120 が存在しない場合（S1402 で「No」）は、ステップ 1401 で検索した結果のエージェントに対して同時時間帯のスケジュールの再設定を行う。再設定では、同時時間帯にスケジュールされたジョブを、ジョブテーブル 400 の優先度 405 に基づいて、優先度順に並び替える（S1403）。次に、遅延が発生するジョブより優先度が低い、若しくは同じジョブがある場合（S1404 で「Yes」）は、低優先度に対するジョブの再スケジューリングを行う（S1405）。ここで遅延が発生しないジョブの再スケジューリングが行えた場合（S1406 で「Yes」）は、スケジュールテーブル 300 を変更し、ジョブの実行エージェントを変更する（S1411）。

【0068】

再スケジューリングにより遅延が発生する、若しくはジョブの再スケジューリングを行う事ができない場合は、遅延が発生するジョブより高い優先度をもつジョブに対して、再スケジューリングの検索を行う。最初に遅延が発生するジョブより高い優先度をもつジョブが存在するかを確認し、存在する場合（S1407 で「Yes」）は、検索処理を行う（S1408）。検索処理の結果、遅延が発生しないスケジュールが検索できた場合（S1409 で「Yes」）ジョブを置き換えるため、スケジュールテーブル 300 を変更し、ジョブの実行エージェントを変更する（S1411）。遅延が発生しないスケジュールが検索できない場合（S1409 で「No」）は、低優先度検索処理 1405 の結果から遅延の最小となるジョブの置き換えを採用し、スケジュールテーブル 300 を再設定し（S1410）、コンソール 103 にワーニングメッセージを出力する（S1412）。

【0069】

また代替となるエージェント 120 が 1 つもない場合は、エラーメッセージを

コンソール 103 に出力する (S1413)。

【0070】

再スケジューリング部の処理は、ステップ 1400 やステップ 1401 で検索した結果のエージェントに対して再スケジューリング部の処理を行うため、最初の呼び出し以外は再スケジューリング部内部から再起的に呼び出されるようにすることによって、再スケジュールに関連するエージェントに対し、順次再スケジュールを行うことが可能となる。

【0071】

尚、本フローは適宜順序を改変可能であり、実施形態に応じては、ジョブに優先度を設けず、優先度検索処理を省略することも可能である。

【0072】

図 15 は低優先度検索処理 1405 の一実施形態を示すフローチャートである。

【0073】

低優先度検索処理では、置き換えの対象となるジョブを Job1 とし (S1501)、Job1 と優先度が同じ、若しくは低いジョブそれぞれに対して検索処理を行う (S1502～S1512)。最初に、優先度の同じ、若しくは低いジョブの 1 つを Job2 に設定する (S1503)。Job1 と Job2 を置き換えた場合に、Job2 の開始時間に遅れる事なく再スケジュールができる場合は (S1505 で「Yes」)、この組み合わせを採用し、検索処理を終了する (S1513)。

【0074】

遅延が発生する場合 (S1505 で「No」) は、遅延時間を計算する (S1506)。遅延時間の計算は、置き換え後の実行開始日時 301 を求め、その時点から最大実行時間 303 を足すことで、最終終了日時 302 からどのくらい遅延するかで行う。

【0075】

次に、Job2 に対して、再度低優先度検索処理 (高優先度検索処理) を呼び出す (S1507)。これにより再度ステップ 1500 からの処理を再帰的に呼

び出す事で置き換えの対象となるジョブの組み合わせを検索する。

【0076】

ステップ1507の結果、J o b 2が遅延なしで再スケジューリングができる場合（S 1 5 0 8で「Yes」）は検索を終了し、検索結果を再スケジューリングの結果として採用する（S 1 5 1 3）。

【0077】

遅延なしの組み合わせが検索できない場合（S 1 5 0 8で「No」）は、先ほど計算した遅延時間がこれまでの検索結果より小さいかを判定し（S 1 5 0 9）、小さい場合はジョブ置き換えの組み合わせとその遅延時間を保存する（S 1 5 1 1、S 1 5 1 2）。

【0078】

図16は高優先度検索処理1408の一実施形態を示すフローチャートである。

【0079】

高優先度検索処理では、置き換えの対象となるジョブをJ o b 1とし（S 1 6 0 1）、J o b 1と優先度が高いジョブそれぞれに対して検索処理を行う（S 1 6 0 2～S 1 6 0 9）。最初に、優先度の高いジョブの1つをJ o b 2に設定する（S 1 6 0 3）。J o b 1とJ o b 2を置き換えた場合に、J o b 2の開始時間に遅れる事なく再スケジュールができる場合は（S 1 6 0 5で「Yes」）、この組み合わせを採用し、検索処理を終了する（S 1 6 0 8）。

【0080】

遅延が発生する場合（S 1 6 0 5で「No」）は、J o b 2に対して、再度高優先度検索処理（低優先度検索処理）を呼び出す（S 1 6 0 6）。これにより再度ステップ1600からの処理を再帰的に呼び出す事で置き換えの対象となるジョブの組み合わせを検索する。

【0081】

ステップ1606の結果、J o b 2が遅延なしで再スケジューリングができる場合（S 1 6 0 7で「Yes」）は検索を終了し、検索結果を再スケジューリングの結果として採用する（S 1 6 0 8）。遅延なしの組み合わせが検索できない場

合 (S 1607で「No」) は、採用せずに終了する。

【0082】

図17は、マネージャ110における代替エージェント検索部205の処理 (図14のステップ1400における処理) の一例を示すフローチャートである。

【0083】

スケジュールの置き換えを行うジョブの利用するリソースをジョブテーブル400から取得する (S1700)。次にリソース間距離テーブル600から必要な全てのリソースを使用できる実行サーバ101を検索する (S1701)。ここでリソースを使用できる実行サーバ101が存在した場合 (S1702)、実行コストを計算する (S1703)。実行コストは、ジョブテーブル400内の最大許容コスト以下になるようにし (S1704)、そのようなリソースの組み合わせがある実行サーバ101とそこで動作するエージェント120を選択する (S1705)。

【0084】

尚、ステップ1701でリソース間距離テーブルから必要な全てのリソースを使用できる実行サーバを検索する際、複数の実行サーバで使用可能なリソースを組み合わせることにより、複数の実行サーバとそこで動作するエージェントを特定しても良い。

【0085】

図18はジョブのスケジュール再設定の具体例を示す。

【0086】

図18 (a) のようにジョブがスケジュールされていた場合を考える。またこのときのジョブテーブル400とリソース間距離テーブル600はそれぞれ図4と図6であるとする。

【0087】

ジョブ1が終了予定時刻より遅延する (1800)。マネージャ110のジョブ実行中エージェント管理部203は、ジョブ1の遅延を検知し、他のジョブに影響があるかを検索し、次に実行される予定であるジョブ2へ影響がある事を検知する (1801)。

【0088】

ここでマネージャ110は、図14で述べた方法を用いて、ジョブ2の再スケジューリングを行う。ジョブ2を実行するためには、CPUとメモリとディスクを必要とする(図4)。ジョブ2を実行サーバ2と実行サーバ3で実行可能であるが(図6)、最大許容コストを計算すると実行サーバ2は15、実行サーバ3は25であるため実行サーバ2のみ代替可能である(図4と図6)。そこで、ジョブ2を実行サーバ2で実行するように再スケジューリングする(1802、1803)。しかし同時刻に実行サーバ2では、ジョブ3がスケジュールされている(1804)。そこでジョブ3のスケジュールの再構成を行う。ジョブ3を実行できる実行サーバは、実行サーバ3のみ(図6)であるため、ジョブ3を実行サーバ3で実行するようにする(1805、1806)。しかしジョブ4が同時刻にスケジュールされているため(1807)、ジョブ4の再スケジューリングが必要となる。ジョブ4を実行できる実行サーバは実行サーバ1のみ(図6)である。実行サーバ1は、本来ならばジョブ2の実行がスケジュールされていたが、これまでのスケジュールの再構成により、スケジュールの空きができている(1802)。これによりジョブ4の再スケジュールが終了する(1807、1808)。

【0089】

以上より全てのジョブに対してスケジュールの再構成を行う事ができ、ジョブ1の遅延による影響を最小限に留める事ができる。

【0090】

図19は、優先度を加味した場合のジョブのスケジュール再設定の具体例を示す。

【0091】

図19(a)のようにジョブがスケジュールされていた場合を考える。またこのときのジョブ2は、実行サーバ1と実行サーバ2と実行サーバ3のいずれでも実行可能とし、ジョブ3は、実行サーバ1と実行サーバ2で実行可能、ジョブ4は、実行サーバ1と実行サーバ3で実行可能であるとする。ジョブの優先度は、それぞれジョブ2が優先度B、ジョブ3が優先度A、ジョブ4が優先度Cとし、ジ

ジョブ3が最も優先度が高いものとする。

【0092】

ジョブ1が終了予定時刻より遅延する(1900)。マネージャ110は、ジョブ1の遅延を検知し、他のジョブに影響があるかを検索し、次に実行される予定であるジョブ2へ影響がある事を検知する(1901)。

【0093】

ここでマネージャ110は、図14で述べた方法を用いて、ジョブ2の再スケジューリングを行う。再スケジューリングを行う実行サーバとしては、実行サーバ2、実行サーバ3の二つがあるが、それぞれ同時時間帯にジョブ3とジョブ4がスケジュールされている(1902、1903)。ここでジョブ3と置き換えを考えた場合、ジョブ3が実行サーバ1に移動するが、遅延が発生する(1904)。しかしジョブ3はジョブ2より優先度が高いため、この置き換えを行う事ができない(1905)。

【0094】

ジョブ4との置き換えを行う場合(1906)、ジョブ3同様にジョブ4も遅延するが、ジョブ4の優先度がジョブ2より低いため遅延する場合でも、再スケジューリングを行う(1907)。

【0095】

またこの際コンソール103にワーニングメッセージを出力し、遅延が発生する事を管理者に知らせる(1908)。

【0096】

以上説明したように、本実施形態によれば、ジョブを実行する計算機で障害や性能の劣化が発生した場合に他の計算機や他のジョブのスケジュールを考慮したジョブのスケジュールを再構成する技術を提供可能となる。また、利用可能なリソースの追加や削除に柔軟に対応でき、管理者がジョブ実行のスケジュールの設定を変更せずとも、常にリソースを有効に利用したジョブ処理と運用管理を行うことが可能となる。

【0097】

上述した実施形態は、本発明の主旨を逸脱しない範囲で適宜変更や組み合わせ

可能である。

【0098】

【発明の効果】

本発明によれば、リソースの稼動状況に応じたジョブスケジュールを適宜柔軟に提供可能となる。

【図面の簡単な説明】

【図1】

ジョブを実行するシステム構成の一実施形態を示す全体構成図

【図2】

マネージャの主要機能の一実施形態を示す機能ブロック図

【図3】

スケジュールテーブルのデータ構成の一例を示す図

【図4】

ジョブテーブルのデータ構成の一例を示す図

【図5】

エージェント管理テーブルのデータ構成の一例を示す図

【図6】

リソース間距離テーブルのデータ構成の一例を示す図

【図7】

マネージャの基本処理の一例を示すフローチャート

【図8】

エージェントの主要機能の一実施形態を示す機能ブロック図

【図9】

履歴パフォーマンステーブルのデータ構成の一例を示す図

【図10】

リソース使用制限テーブルのデータ構成の一例を示す図

【図11】

エージェントの基本処理の一例を示すフローチャート

【図12】

リソース使用制限テーブルの更新処理の一例を示すフローチャート

【図 13】

マネージャのジョブ実行中エージェント管理部における処理の一例を示すフローチャート

【図 14】

マネージャの再スケジューリング部における処理の一例を示すフローチャート

【図 15】

低優先度検索処理の一実施形態を示すフローチャート

【図 16】

高優先度検索処理の一実施形態を示すフローチャート

【図 17】

マネージャの代替エージェント検索部の処理の一例を示すフローチャート

【図 18】

ジョブ実行のスケジュールの再構成の一実施形態

【図 19】

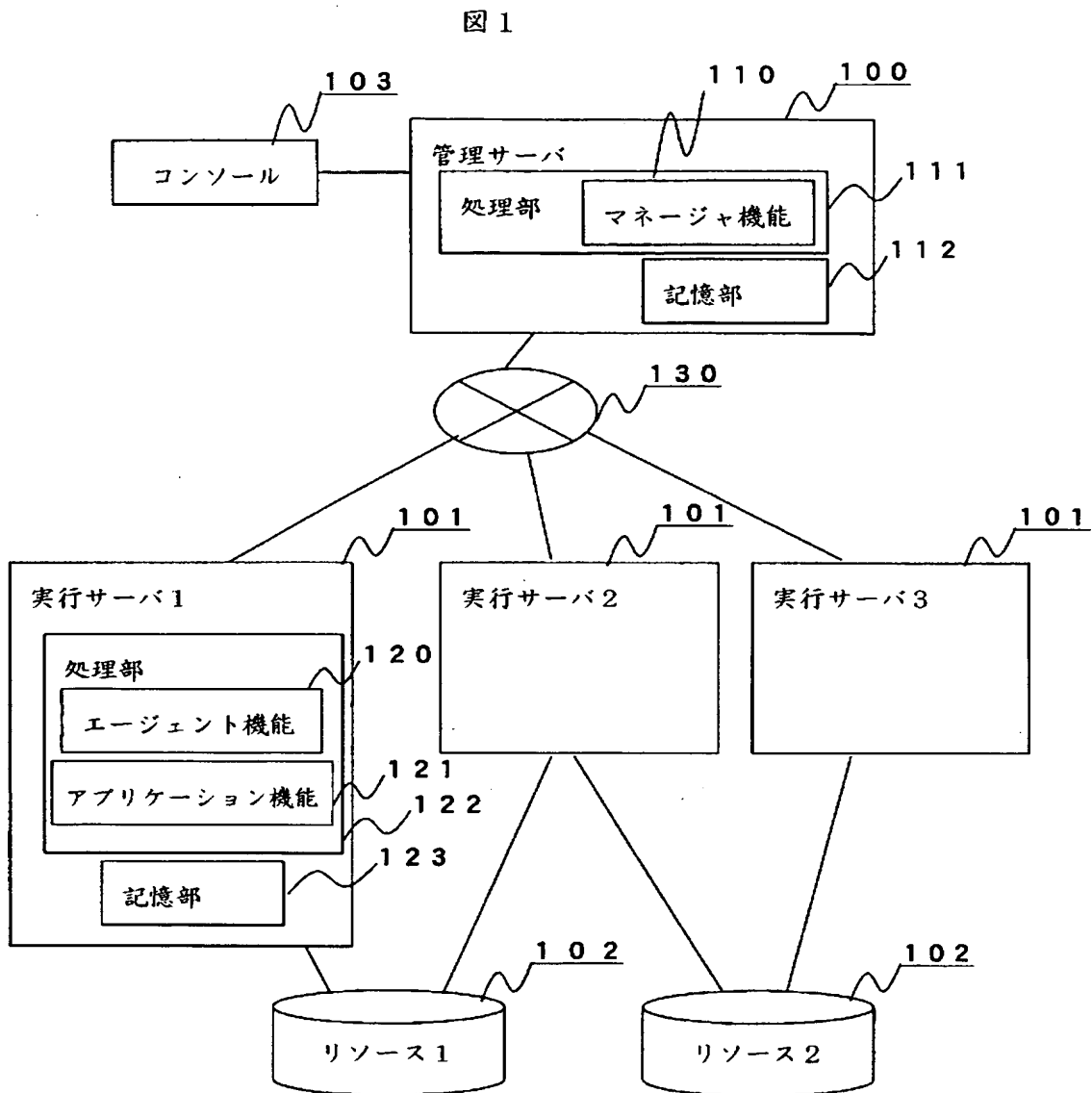
ジョブ実行のスケジュールの再構成の他の実施形態

【符号の説明】

- 100 管理サーバ
- 101 実行サーバ
- 102 リソース
- 103 コンソール
- 110 マネージャ
- 111 処理部
- 112 記憶装置（記憶部）
- 120 エージェント
- 121 アプリケーション
- 122 処理部
- 123 記憶装置（記憶部）

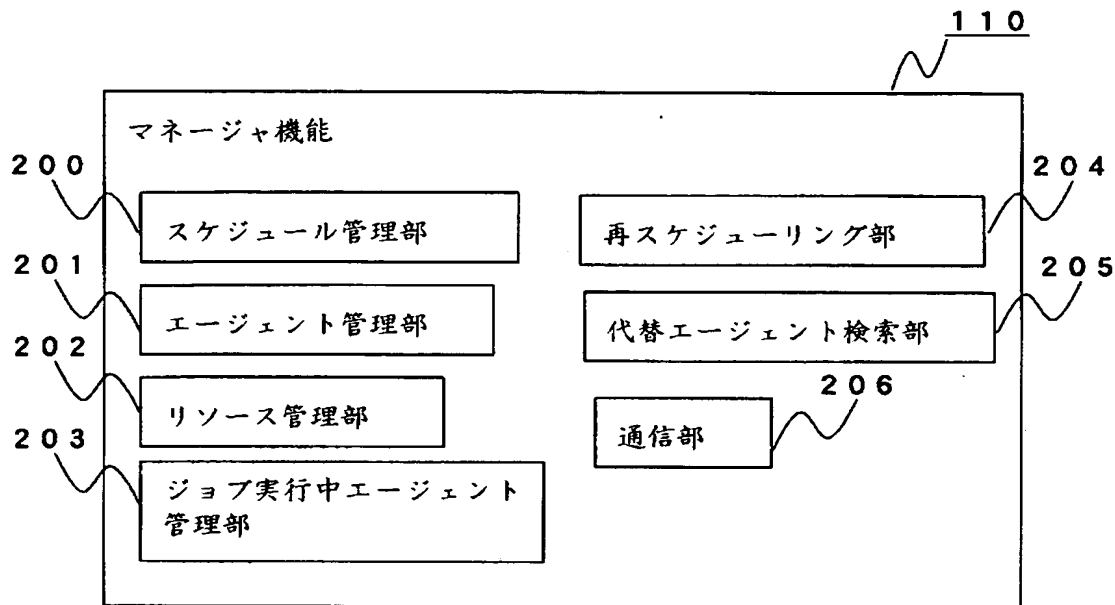
【書類名】 図面

【図1】



【図 2】

図 2



【図 3】

図 3

スケジュールテーブル 300

301 実行開始日時	302 最終終了日時	303 最大実行時間	304 実行エージェ ント	305 実行ジョ ブ
2002/11/10 10:00:00	2002/11/10 11:00:00	01:00	エージェント 1	ジョブ 1
2002/11/10 23:00:00	2002/11/11 00:00:00	00:30	エージェント 3	ジョブ 2
毎週水曜 12:00:00	-	02:00	エージェント 2	ジョブ 3
・	・	・	・	・
・	・	・	・	・
・	・	・	・	・

【図 4】

図 4

ジョブテーブル 400

ジョブ名	実行内容	実行必要リソース	最大許容コスト	優先度
ジョブ 1	計算処理	CPU, メモリ	10	A
ジョブ 2	バッチ処理	CPU, メモリ, ディスク	20	B
ジョブ 3	DB 処理	CPU, メモリ, データベースソフト, リソース 2	70	A
ジョブ 4	DB 処理	CPU, メモリ, データベースソフト, リソース 1	30	C
.
.
.

【図 5】

図 5

エージェント管理テーブル 500

エージェント名	実行サーバ
エージェント 1	実行サーバ 1
エージェント 2	実行サーバ 2
エージェント 3	実行サーバ 3
エージェント 4	実行サーバ 2
.	.
.	.
.	.

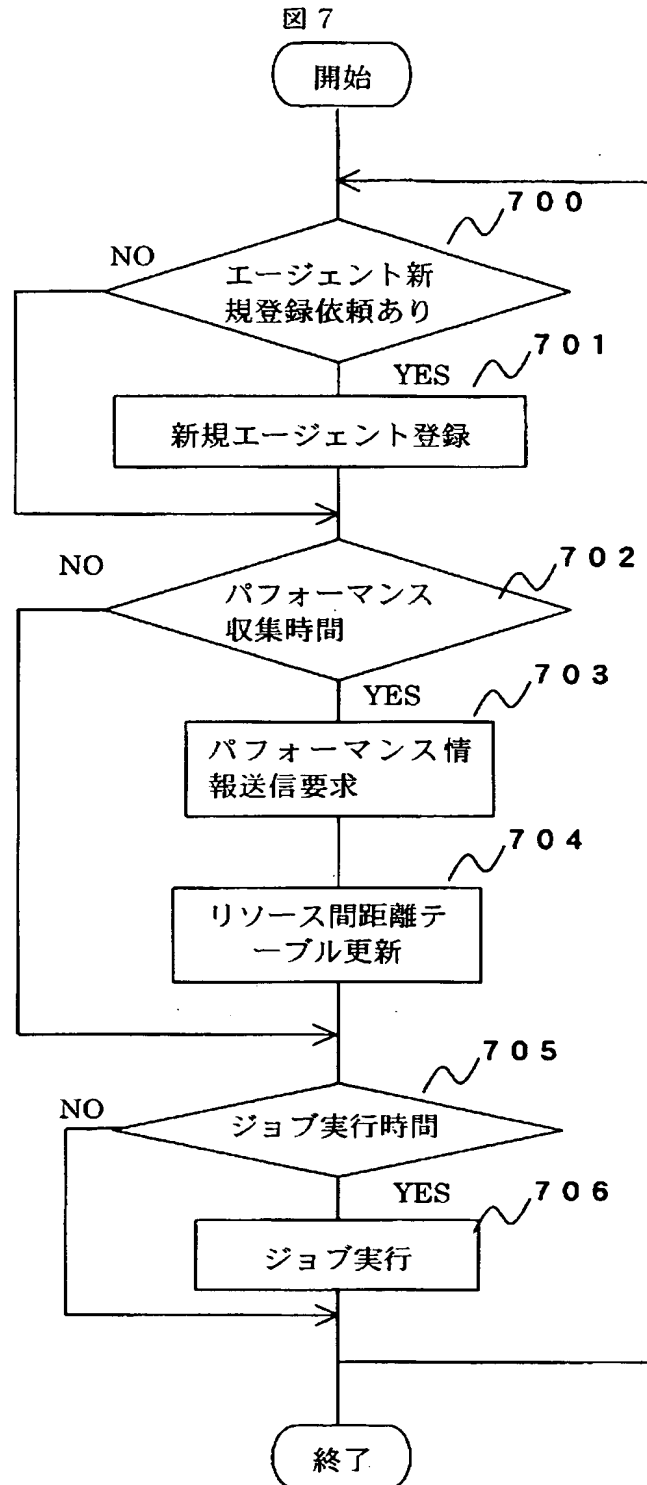
【図6】

図6

リソース間距離管理テーブル 600

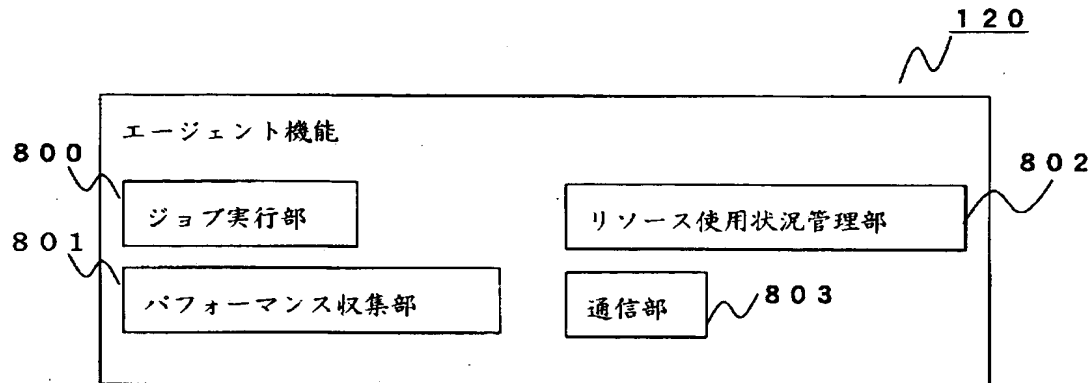
			使用実行サーバ			
			実行サーバ 1	実行サーバ 2	実行サーバ 3	...
使用 可能 リ ソ ー ス	実行 サーバ1	CPU	5	-	-	
		メモリ	5	-	-	
		ディスク	5	15	-	
		データベ ース	10	20	-	
		...				
	実行 サーバ2	CPU	-	5	-	
		メモリ	-	5	-	
		ディスク	-	5	15	
		...				
	実行 サーバ3	CPU	-	-	10	
		メモリ	-	-	5	
		ディスク	15	20	10	
		データベ ース	15	30	10	
		...				
	リソ ース1	ディスク	10	-	40	
	リソ ース2	ディスク		20	40	

【図 7】



【図 8】

図 8



【図 9】

図 9

履歴パフォーマンステーブル 900

収集時刻	収集項目					...
	CPU %	使用メモリ量	空きディスク容量	平均ディスク処理時間	平均クエリ処理時間	
10:00:01	30%	200MB	1000MB	0.5ms	0.01ms	
10:00:02	23%	256MB	900MB	0.2ms	0.02ms	
10:00:03	10%	128MB	900MB	0.6ms	0.05ms	
...						
11:00:00	40%	200MB	500MB	1.2ms	0.25ms	
11:00:01	14%	250MB	700MB	0.3ms	0.10ms	
.
.
.

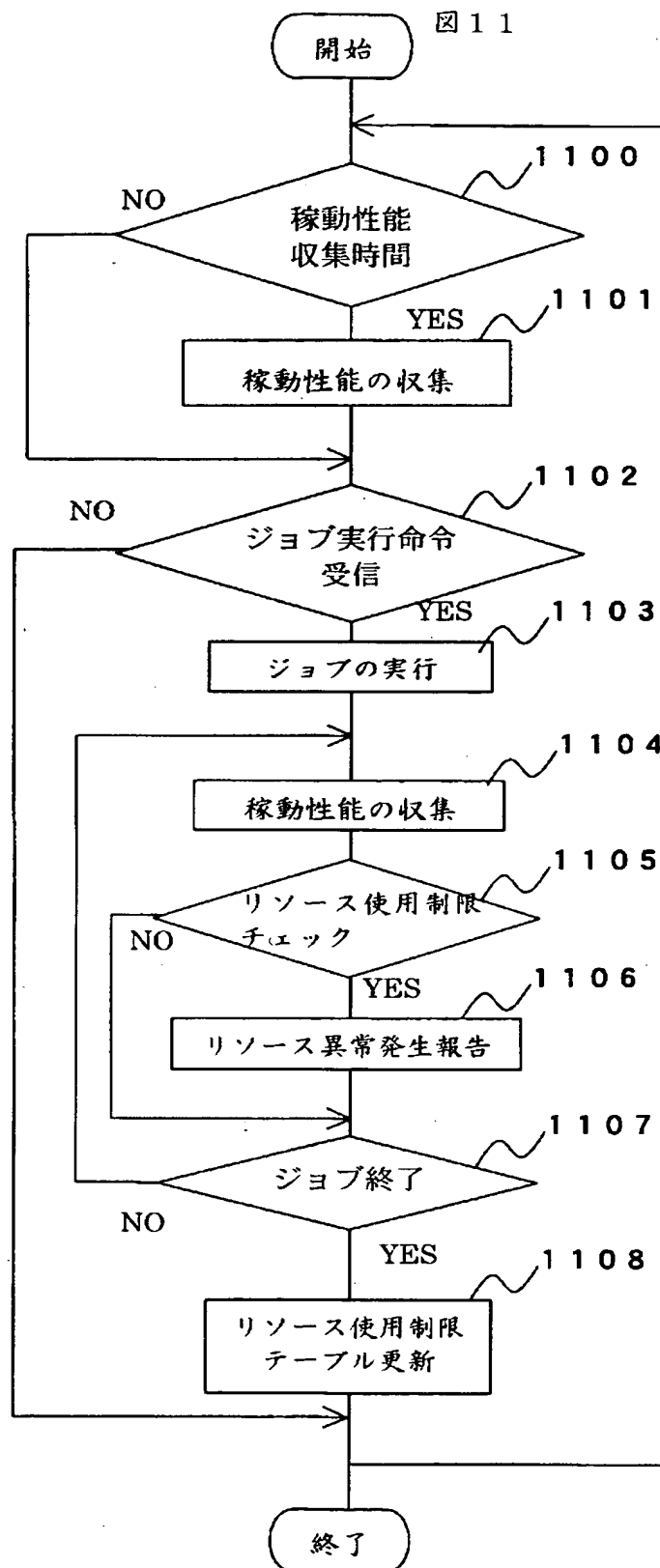
【図 10】

図 10

リソース使用制限テーブル 1000

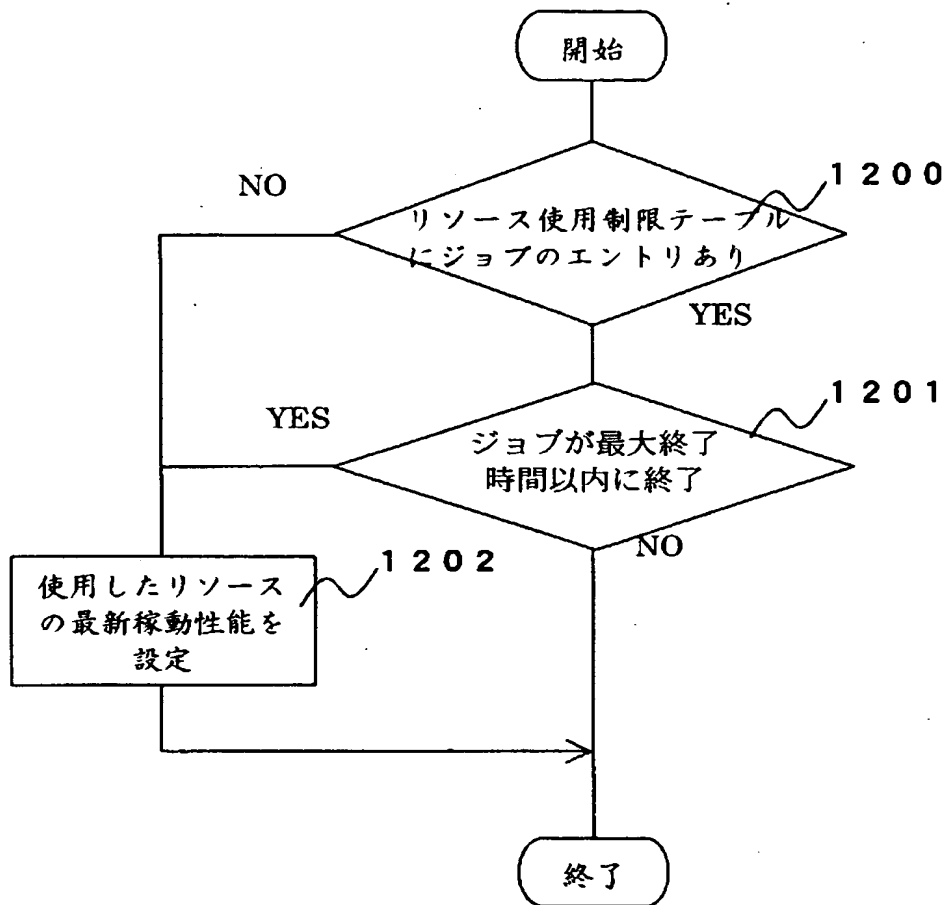
ジョブ	しきい値 越え回数	しきい値			
		CPU 使用率	使用メモリ量	空きディスク容 量	...
ジョブ 1	2/3 回	50%以上	384MB 以上	200MB 以下	
ジョブ 2	2/3 回	60%以上	-	300MB	
ジョブ 3	1/1 回	80%以上	-	.	

【図 11】

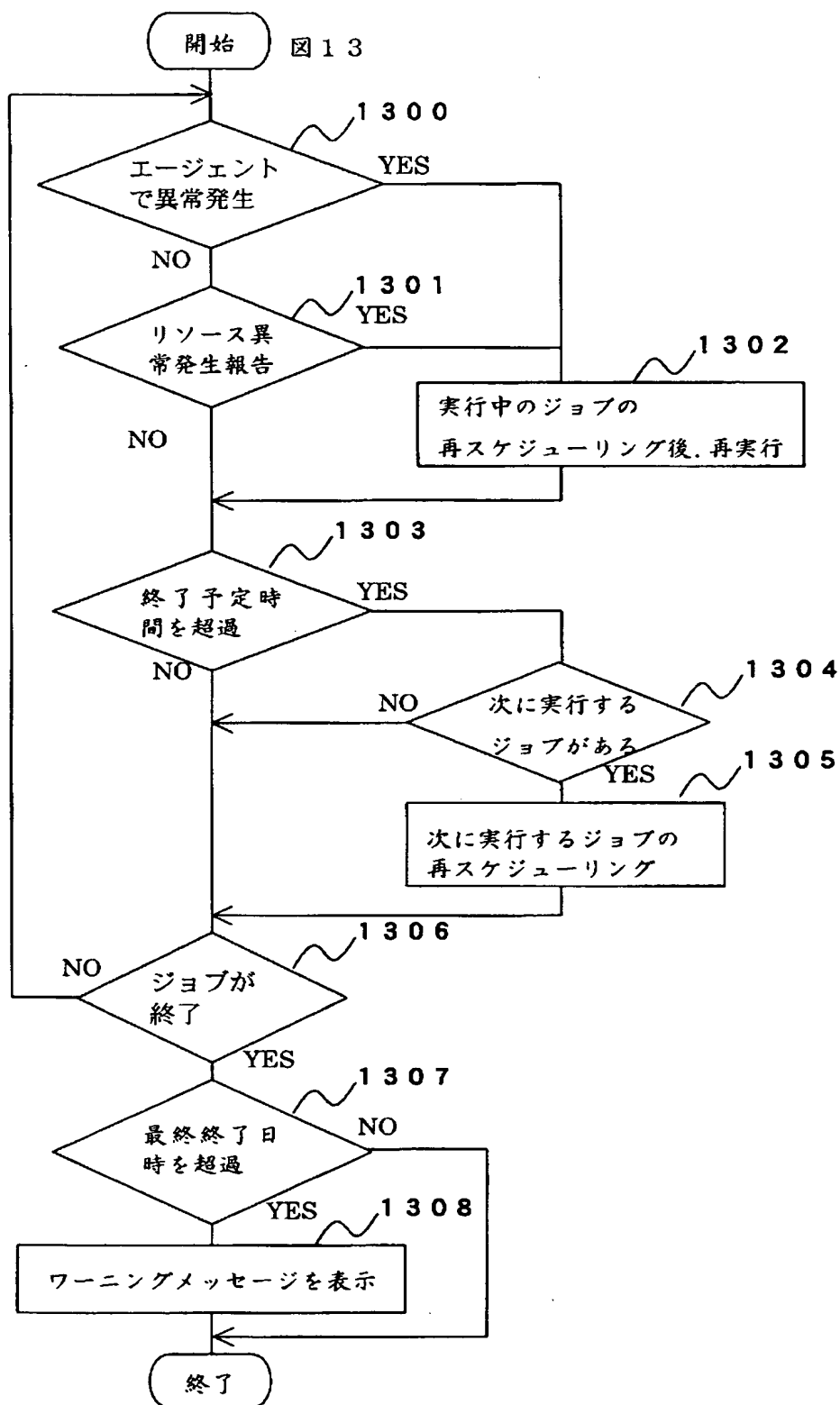


【図 12】

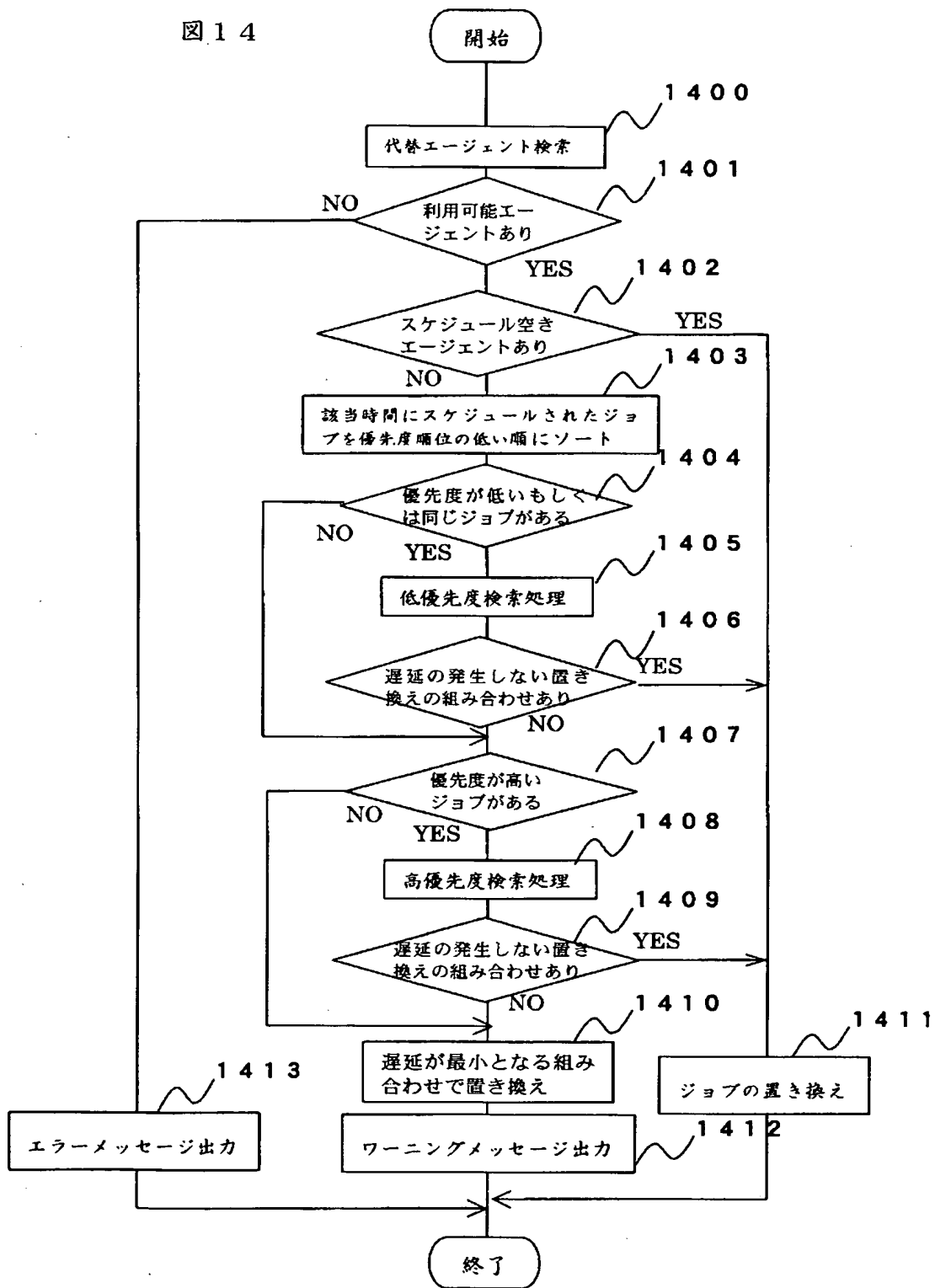
図 12



【図 13】

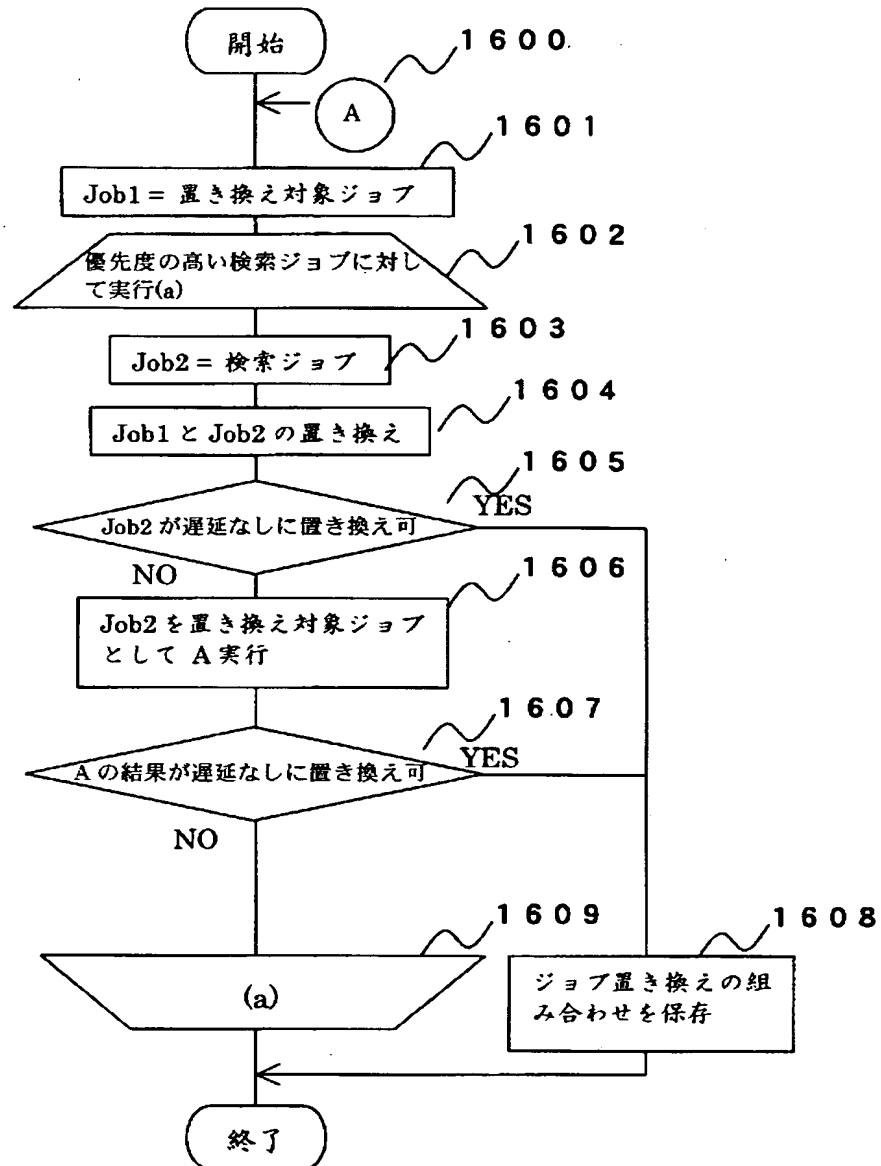


【図 14】



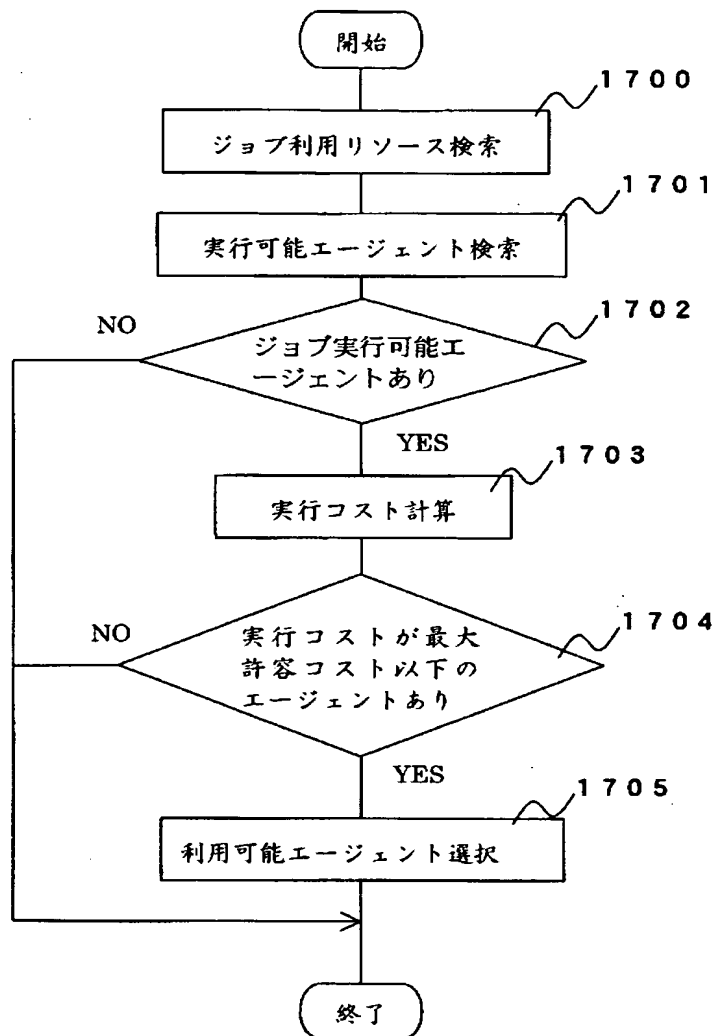
【図 16】

図 16

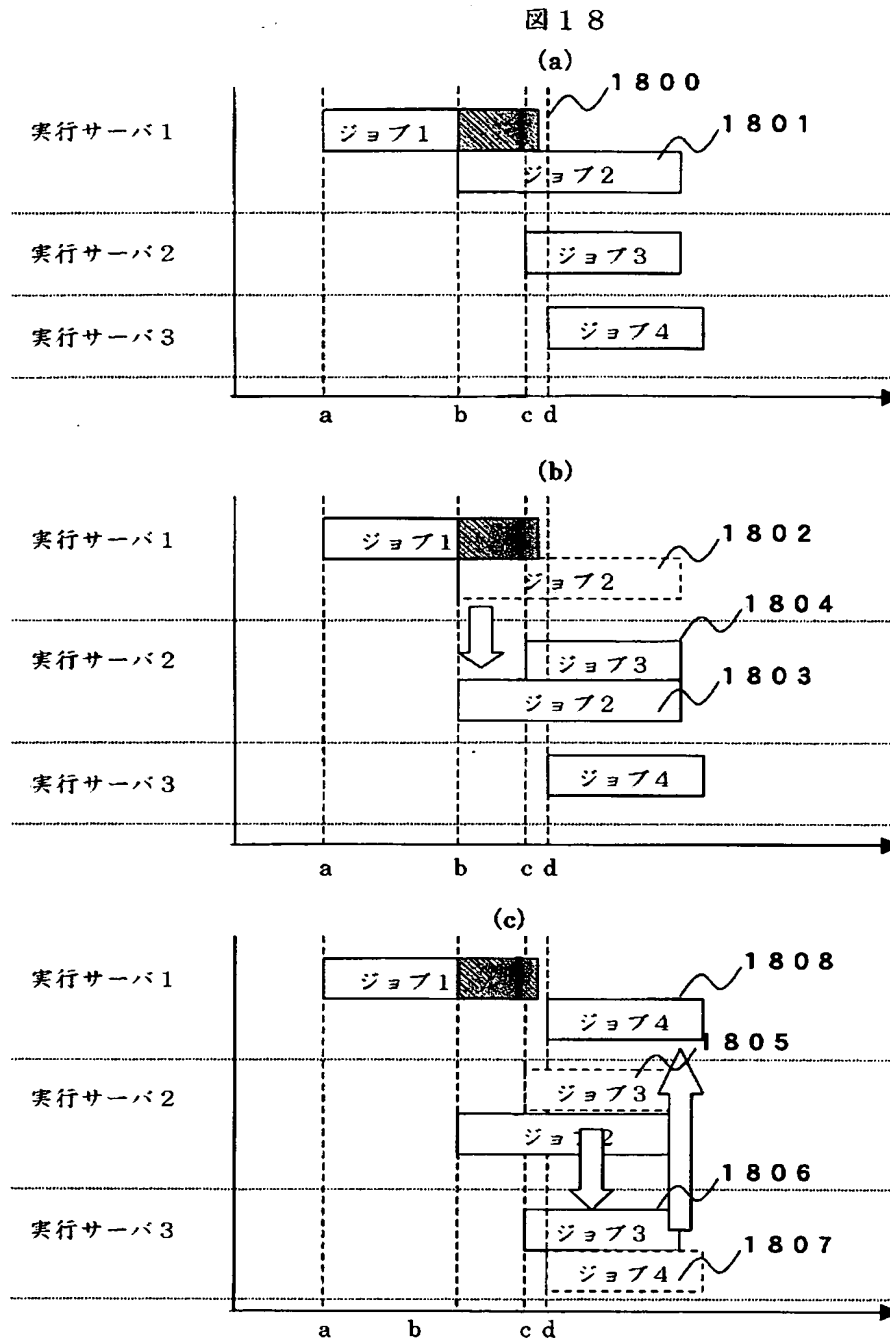


【図 17】

図 17

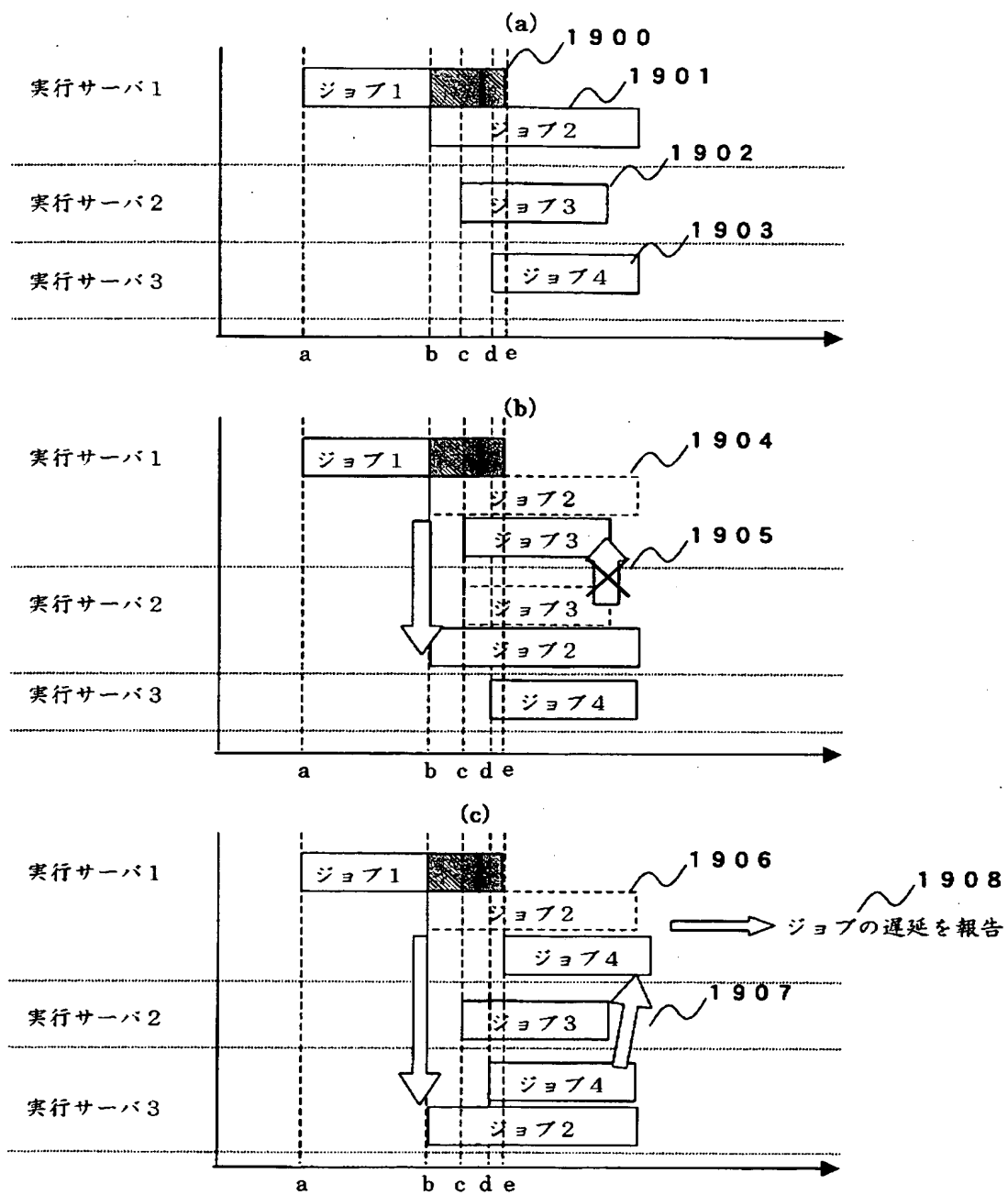


【図18】



【図 19】

図 19



【書類名】 要約書

【要約】

【課題】

ジョブを実行する実行サーバで障害や性能の劣化が発生した場合にジョブのスケジュールを再構成する技術を提供する。

【解決手段】

ジョブのスケジュールを管理するジョブスケジュール管理方法であって、ジョブが割り当てられた計算機の稼動状況を監視し、稼動状況が所定の条件を満たすか否か判断し、稼動状況が所定の条件を満たす場合、計算機に割り当てられたジョブのうち、所定の条件を満たした時点で未完了のジョブを検出し、検出した未完了のジョブの実行に必要なリソース情報に基づいて、検出した未完了のジョブを実行可能な他の計算機を抽出し、検出した未完了のジョブを抽出した他の計算機に割り当てる。

【選択図】 図1

認定・付加情報

特許出願の番号	特願 2003-193259
受付番号	50301130125
書類名	特許願
担当官	第七担当上席 0096
作成日	平成15年 7月 9日

<認定情報・付加情報>

【提出日】 平成15年 7月 8日

特願 2003-193259

出願人履歴情報

識別番号

[000005108]

1. 変更年月日

1990年 8月31日

[変更理由]

新規登録

住 所

東京都千代田区神田駿河台4丁目6番地

氏 名

株式会社日立製作所